Napredni operacijski sustavi -- dodatni materijali i upute

Skripta s tekstom o ulazno-izlaznim napravama i višeprocesorskim sustavima

Ulazno-izlazne naprave - LAB2

1. Primjeri koda koji se koriste u skripti i na vježbama

- Kod na githubu

2. Upute za drugu laboratorijsku vježbu

Druga vježba se sastoji od tri zadatka. Prva dva donose (maksimalno) po tri boda, a treći četiri boda (ovisno o tjednu kad se predaju).

Sadržaj vježbi biti će detaljnije pojašnjen na predavanjima (upute su u skripti).

Vježbe bi trebalo moći izraditi u bilo kojoj distribuciji Linuxa. Međutim, obzirom da se zadire u jezgru OS-a i puno toga može poći krivo, preporučeno je koristiti pripremljenu sliku (Ubuntu.20.04.64-bit.7z) za VMware okruženje. Asistenti mogu (ponekad) pomoći ali samo ako se koristi ovaj sustav.

2.1. Lab2a: Korištenje operacije `poll`

Napisati dva programa koji će koristiti kod iz direktorija `lab2a`, ili neku njegovu modifikaciju.

Modul pokretati tako da se stvori više naprava i međuspremnika (može jednako).

Prvi program otvara sve naprave za čitanje, s `poll` čeka da se na bilo kojoj pojavi znak, čita ga i ispisuje.

Drugi program otvara sve naprave za pisanje te periodički (npr. svakih 5 sekundi) s `poll` provjerava je li barem jedna od njih spremna za prihvat novih znakova i ako je nasumice odabire jednu takvu i šalje joj jedan znak.

2.2. Lab2b: Korištenje operacije `ioctl`

Napraviti modul koji će stvoriti tri naprave: ulaznu, radnu i izlaznu (koristiti skoro gotov kod u `lab2b`).

Ulazna ostvaruje samo operacije `open` i `write` - dobivene podatke sprema u svoj međuspremnik.

Izlazna naprava ostvaruje samo operacije `open` i `read` - šalje podatke iz svog međuspremnika.

Radna naprava ostvaruje samo `open` i `ioctl`. Kad joj se s `ioctl` pošalje naredba, tj. broj, prebacuje zadani broj bajtova iz međuspremnika ulazne naprave u međuspremnik izlazne, ili završava ako brojeva više nema u ulaznom međuspremniku ili se izlazni prepuni.

Pri učitavanju modula treba stvoriti i alarm (timer) koji će periodički (npr. svakih pet sekundi) prebacivati jedan znak iz ulaznog međuspremnika u izlazni.

2.3. Lab2c: (opcionalni dio) Mehanizmom naprava ostvariti red poruka ili cjevovod

Studenti s predzanjom parnom znamenkom u JMBAGu trebaju ostvariti red poruka, oni s neparnom cjevovod.

2.3.1. Red poruka

Upravljačkim programom ostvariti jedan red poruka koji istovremeno može koristiti više dretvi, neke za slanje poruka i neke za primanje poruka.

Dretve trebaju moći koristiti mehanizam reda poruka preko sučelja za rad s datotekama (`open`, `close`, `read`, `write`). Slanje poruke obaviti preko sučelja `write` a čitanje preko `read`. Nije potrebno da poruka ima prioritet (ili neku drugu oznaku). Primjerom programa pokazati rad reda poruka (programima koji ga koriste).

Pri pokretanju modula argumentima definirati najveći broj poruka u redu, najveću veličinu poruke i najveći broj procesa/dretvi koje mogu istovremeno raditi s redom (pozvati `open`).

Pri otvaranju reda od strane nekog procesa (funkcija `open`) mora se koristiti zastavica `O_RDONLY` ili `O_WRONLY`. Za sve ostale zastavice javiti grešku. Ako je korištena zastavica `O_RDONLY` onda operacija `write` mora javiti grešku, i obratno, ako je korištena zastavica `O_WRONLY` onda read mora javiti grešku.

Funkcija `write` u red stavlja jednu poruku zadane veličine. Ukoliko je veličina podataka koji se šalje veći od maksimalno definirane za red, javiti grešku. Ukoliko je red poruka već pun (već ima maksimalan broj poruka u redu), onda blokirati dretvu dok se neka poruka ne pročita. Pri uspješnom stavljanju poruke u red, odblokirati prvu iz reda dretvi koje čekaju na poruku (npr. koristiti semafor).

Funkcija `read` čita jednu poruku iz reda. Ako je veličina poruke koju se želi pročitati manja od veličine najveće poruke koja stane u red, javiti grešku. Ako je red prazan, blokirati dretvu. Pri uspješnom uzimanju poruka iz reda, odblokirati prvu dretvu koja želi staviti poruku u red.

2.3.2. Cjevovod

Upravljačkim programom ostvariti jedan cjevovod koji istovremeno može koristiti više dretvi, neke za stavljanje podataka u cijev i neke za uzimanje podataka.

Dretve trebaju moći koristiti mehanizam cjevovoda preko sučelja za rad s datotekama (`open`, `close`, `read`, `write`). Stavljanje podataka u cjevovod obaviti preko sučelja `write` a čitanje preko `read`. Primjerom programa pokazati rad cjevovoda (programima koji ga koriste).

Pri pokretanju modula argumentima definirati veličinu kružnog međuspremnika za cjevovod i najveći broj procesa/dretvi koje mogu istovremeno raditi s njim (pozvati `open`). Preporuka je koristiti postojeće kfifo* sučelje za međuspremnik.

Pri otvaranju cjevovoda od strane nekog procesa (funkcija `open`) mora se koristiti zastavica `O_RDONLY` ili `O_WRONLY`. Za sve ostale zastavice javiti grešku. Ako je korištena zastavica `O_RDONLY` onda operacija `write` mora javiti grešku, i obratno, ako je korištena zastavica `O_WRONLY` onda read mora javiti grešku.

Funkcija `write` u cijev stavlja jedan podatak zadane veličine. Ukoliko je podatak koji se želi staviti prevelik, veći od veličine međuspremnika cjevovoda, javiti grešku. Ukoliko se podatak ne može staviti jer trenutno nema dovoljno praznog prostora u međuspremniku, onda blokirati dretvu dok se ne napravi mjesta. Ukoliko ima blokiranih dretvi, svi ostale dretve koje pozovu `write` treba također blokirati - treba osigurati da neka dretva "kasnije" ne stavi podatak u cijev prije neke druge dretve koja za to vrijeme čeka. Pri uspješnom stavljanju podataka u cjevovod, odblokirati prvu iz reda dretvi koje čekaju na čitanje (npr. koristiti semafor).

Funkcija `read` čita jedan podatak iz cjevovoda. Ukoliko je međuspremnik prazan, blokirati dretvu dok se u cijev nešto ne stavi. Ako ima podataka u cjevovodu, pročitati traženi broj okteta ili manje ako ih toliko nema. Povratnom vrijednošću javiti koliko je podataka pročitano u redu. Npr. u cjevovodu je 10 okteta a `read` traži 50 - pročitati 10 i vratiti tu vrijednost (ne blokirati dretvu na `read` ako se može nešto pročitati). Pri uspješnom uzimanju podataka iz cijevi, odblokirati prvu dretvu koja želi staviti podatke u cijev, ako sada ima dovoljno mjesta u cijevi za njene podatke.

2.4. Linkovi

- Kratke upute za pripremu razvojne okoline
- Linux Device Drivers, Third Edition
- The Linux driver implementer’s API guide
- Unreliable Guide To Locking
- Linux Kernel Teaching